/skill-up

How to Make Claude Work Your Way

Vijay Kodam

Principal Engineer, Nokia | AWS Community Builder

Claude Code Meetup | March 2026

```
my-skill/
  SKILL.md
  scripts/
  references/
  templates/
```

ABOUT ME

Vijay Kodam

Principal Engineer @ Nokia

AWS Community Builder

Working with Kubernetes and AWS since 2016

Claude Code user from Day 1

AI Builder, Blogger, Public Speaker

Connect on LinkedIn

linkedin.com/in/vijaykodam

vijay.eu

Blog • Technical articles • Projects

Disclaimer: Presenting in my personal capacity. Not affiliated with my employer.

The Repetition Tax

Without Skills

"Use our team's coding standards..."

"Format PRs like we discussed..."

"Run linting before committing..."

"Follow our K8s naming conventions..."

"Remember, we use pytest not unittest..."

With Skills

Claude already knows.

Define it once.

Claude applies it every time.

Across every session.

CORE CONCEPT

What Are Agent Skills?

A folder with a SKILL.md file + optional scripts & resources that teach Claude how to do something specific.

Model-Invoked
Claude decides when to use a skill based on task context

Progressive Disclosure
Only loads what's needed: description > instructions > resources

Executable
Can include Python, Bash, Node.js scripts Claude can run

Think of it as...

An onboarding doc for your AI teammate

It reads the manual, so you don't have to repeat yourself.

The Extensibility Stack

Mechanism	What It Does	Analogy	Context Loading
CLAUDE.md	Project-level instructions & conventions	*README for Claude*	Always loaded at session start
MCP Servers	Connect external tools & APIs (GitHub, DBs, Slack...)	*USB ports*	Tool list at start; results when invoked
Skills	Procedural knowledge, workflows & best practices	*Training manuals*	Description at start (~50 tok); full SKILL.md on activation; resources on demand
Hooks	Automated triggers at lifecycle events (pre/post tool use)	*Git hooks*	Never in context — runs externally, passes JSON in/out
Plugins	Shareable bundles (commands + skills + hooks + MCP configs)	*npm packages*	Depends on contents — merges all of the above

MCP gives Claude hands (tools). Skills give Claude expertise (knowledge). Use them together.

Agent Skills = Open Standard

agentskills.io

Write once, use everywhere.
Open format maintained by Anthropic.
Like MCP — but for knowledge.

Same playbook as MCP:

Build the infrastructure,
make it open,
win by being the best at using it.

Already adopted by:

Claude Code	**Cursor**	**VS Code / Copilot**	**OpenAI Codex**
Google Gemini CLI	**DataBricks**	**GitHub**	**Snowflake**

Anatomy of a Skill

SKILL.md

```
---
name: explain-code
description: Explains code with
  diagrams and analogies.
---

# Explain Code

When explaining code:
1. Start with an analogy
2. Draw ASCII diagrams
3. Walk through step-by-step
```

Progressive Disclosure

1 name + description

~50 tokens • always loaded

▼

2 Full SKILL.md instructions

<5k tokens • on activation

▼

3 Scripts & resources

as needed • on demand

Where Skills Live

Personal Skills

`~/.claude/skills/`

Your workflows, available across all projects

Project Skills

`.claude/skills/`

Team workflows, committed to git, shared automatically

Plugin Skills

`via /plugin install`

Bundled with installed plugins from marketplaces

Claude.ai / API

`Settings > Skills` or `API container.skills`

Upload custom skills or use Anthropic's built-in ones

LIVE DEMO

Skills in Action

Demo: "Presentation about Presentations"

```
> Create a 3-slide presentation about "Why Skills Matter":
  Title slide, Before vs After comparison,
  Three key takeaways. Save as skills-demo.pptx and open it.
```

What happens behind the scenes:

1. Claude reads the pptx skill description and auto-activates it
2. Skill loads SKILL.md instructions → routes to PptxGenJS guide
3. Claude installs dependencies, writes & runs Node.js code
4. Professional .pptx file generated — from one natural language prompt

One prompt. One skill. Professional output. No repetition.

Best Practices

Description Is Everything

It's how Claude discovers your skill. Include what the skill does AND when to use it. Be specific with trigger phrases.

Keep Skills Focused

One skill = one capability. "PDF form filling" not "document processing". Split broad skills into specific ones.

Security Matters

Skills can execute code. Only use skills from trusted sources. Treat them like npm packages — convenient but verify.

Watch the Context Budget

Skill descriptions share 2% of context window. Too many skills? Run /context to check. Set SLASH_COMMAND_TOOL_CHAR_BUDGET to override.

Debug & Test

Use claude --debug to see skill loading errors. Test with prompts that match your description. Iterate based on real usage.

Key Takeaways

1. Skills turn Claude from a general assistant into a domain specialist

2. MCP = tools, Skills = knowledge, Plugins = distribution — use them together

3. Agent Skills is an open standard — invest in skills and they're portable everywhere

Resources

`code.claude.com/docs/en/skills`	Official Claude Code Skills docs
`agentskills.io`	Open standard specification
`github.com/anthropics/skills`	Anthropic's skills repository